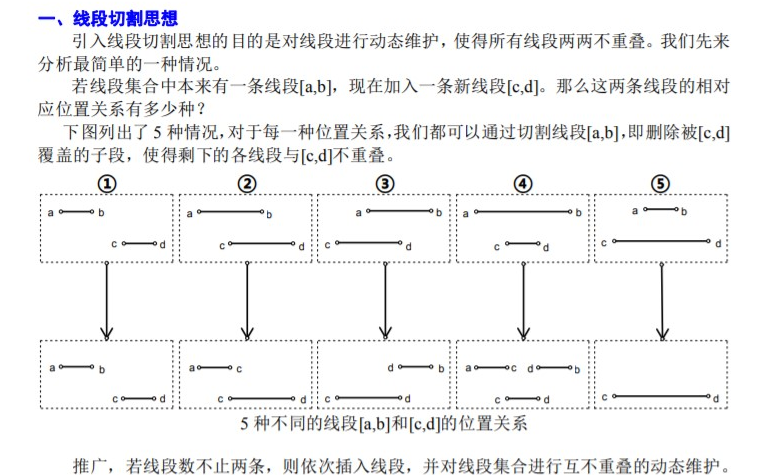
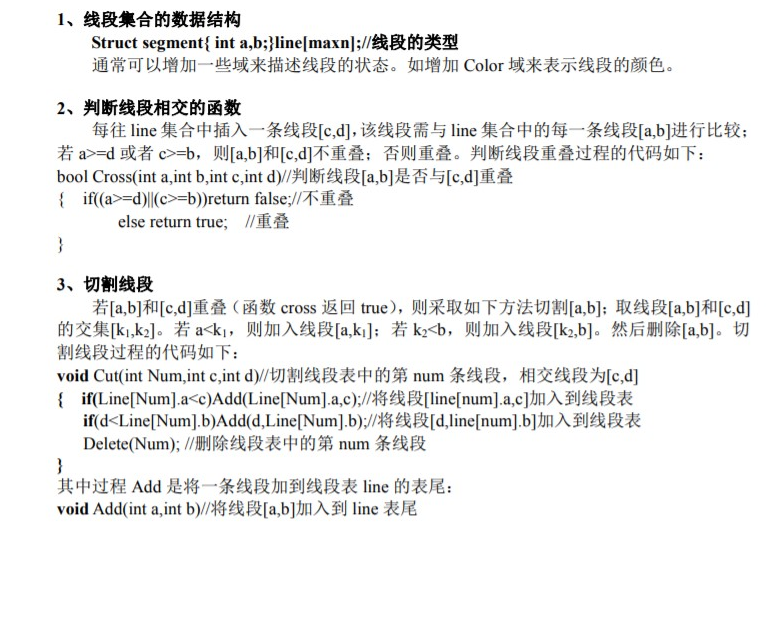
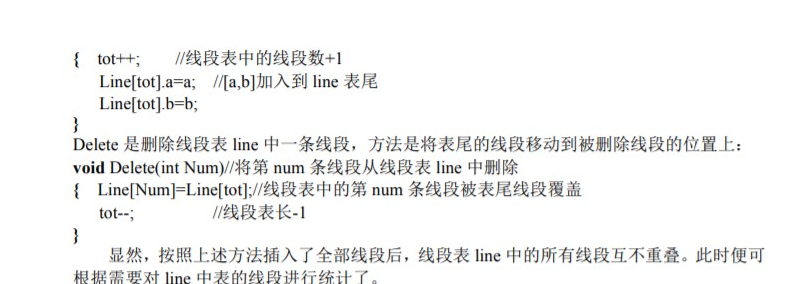
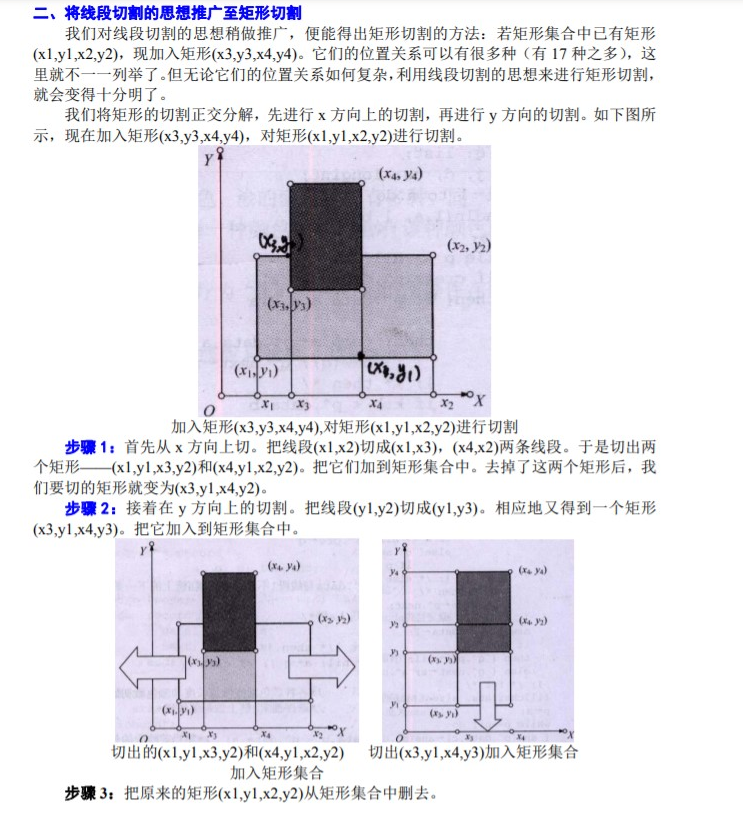
矩形切割只适用于一些数据相对较为简单的题目，可以使代码简单一点，复杂度为o（n\*n）

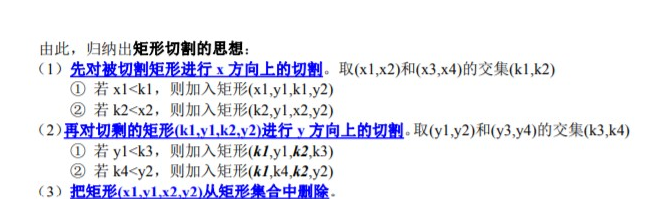
并且还有一定级别的常数，数据量大了可能还是要用线段树+离散化来解决，比如说二维线段树和扫描线











例题：

n 个不同颜色且不透明的长方形被放在一张宽为 aa 长为 bb 的白纸上。它们的边于白纸的边缘平行，且所有的长方形都放置在白纸内。

现在将他们重叠，重叠后会出现不同形状的各种颜色，你需要求出每种颜色的面积。

白纸的左下角的坐标为原点 (0,0)(0,0)，且坐标轴平行于白纸边缘。（注：颜色 1 和底部白纸的颜色相同）

代码：

#include<bits/stdc++.h>

#define ll long long

#define inf 1e18

#define mn 100005

#define y1 y3

using namespace std;

ll tot=0,s[2505]={0},c=0,ans=0,cnt=0,ge,r,n,m;

struct w{ll x1,x2,y1,y2,c;}a[100005];

void add(ll x1,ll y1,ll x2,ll y2,ll c)

{

tot++;

a[tot].x1=x1;

a[tot].x2=x2;

a[tot].y1=y1;

a[tot].y2=y2;

a[tot].c=c;

}

void cut(ll v,ll x1,ll y1,ll x2,ll y2,ll cmd)//矩形切割

{ll k1,k2;

if(cmd==0)//进行 x 方向上的切割

{

k1=max(x1,a[v].x1);

k2=min(x2,a[v].x2);

if(a[v].x1<k1)add(a[v].x1,a[v].y1,k1,a[v].y2,a[v].c);

if(a[v].x2>k2)add(k2,a[v].y1,a[v].x2,a[v].y2,a[v].c);

cut(v,k1,y1,k2,y2,1);

}

if(cmd==1)//进行 y 方向上的切割

{

k1=max(y1,a[v].y1);

k2=min(y2,a[v].y2);

if(a[v].y1<k1)add(x1,a[v].y1,x2,k1,a[v].c);

if(a[v].y2>k2)add(x1,k2,x2,a[v].y2,a[v].c);

}

}

int main()

{

ll cor,x1,x2,y1,y2,x,y,t,i,j,k,z;

char ch;

cin>>n>>m>>t;

add(0,0,n,m,1);

for(i=1;i<=t;i++)

{

scanf("%lld%lld%lld%lld%lld",&x1,&y1,&x2,&y2,&cor);

c=max(c,cor);

for(j=1;j<=tot;j++)//对队列里面的 tot 个矩形进行分别处理

{

if(a[j].x1>=x2||a[j].y1>=y2||a[j].x2<=x1||a[j].y2<=y1)continue;

cut(j,x1,y1,x2,y2,0);//与第 j 个矩形相交，则进行矩形切割

a[j]=a[tot];//最后一个放到前面

tot--;

j--;

}

add(x1,y1,x2,y2,cor);//添加新加入的矩形

}

for(i=1;i<=tot;i++)

s[a[i].c]+=(a[i].x2-a[i].x1)\*(a[i].y2-a[i].y1);//累加颜色为 a[i].c 的面积

for(i=1;i<=c;i++)

if(s[i]>0)printf("%lld %lld\n",i,s[i]);

return 0;

}